DOCUMENT RESUME

ED 313 015 IR 014 065

AUTHOR Grandbastien, Monique; Morinet-Lambert, Josette
TITLE The Teaching and Learning Environment SAIDA: Some Features and Lessons.
PUB DATE 2 Mar 89
NOTE 8p.
PUB TYPE Reports - Descriptive (141)

EDRS PRICE MF01/PC01 Plus Postage.
DESCRIPTORS *Computer Assisted Instruction; Computer Science Education; *Computer System Design; *Expert Systems; Foreign Countries; Higher Education; Instructional Design; Problem Solving; *Programing
IDENTIFIERS France; *Intelligent Tutoring Systems

ABSTRACT

Written in ADA language, SAIDA, a Help System for Data Implementation, is an experimental teaching and learning environment which uses artificial intelligence techniques to teach a computer science course on abstract data representations. The application domain is teaching advanced programming concepts which have not received much attention from developers of intelligent tutoring systems, such as data abstraction and efficient implementation choices. The first part of the paper explains why a knowledge base approach is particularly suitable for this domain and briefly describes the architecture of the system. Part two presents a typology of the system's users and outlines the various SAIDA functions provided for each kind of user. The third part describes the importance of a meaningful structure of the knowledge domain and discusses several issues which emerge from the construction of knowledge bases for educational purposes. (5 references) (GL)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

Monique GRANDBASTIEN Josette MORINET LAMBERT
March 2, 1989

U.S. DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)

☒ This document has been reproduced as received from the person or organization originating it.

☐ Minor changes have been made to improve reproduction quality.

• Points of view or opinions stated in this document do not necessarily represent official OERI position or policy.

ED313015

Abstract :

SAIDA is a teaching and learning environment using a knowledge based system. The application domain is teaching advanced programming concepts such as data abstraction and efficient implementation choices, which has not benefit of many efforts from ITS developers. It leads to define a learning environment for non-beginners. Its main characteristics are to be an open environment and to allow student initiative. The system's design enables to identify numerous actors adding new components to previous described ITS models. The functionalities provided by SAIDA to these different actors are described and several issues to build knowledge bases for educational purposes are discussed.

Key-words :
Knowledge Base System, Computer Education, Data Abstraction

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY
J. Morinet Lambert
TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

IR014045

BEST COPY AVAILABLE

The teaching and learning environment SAIDA : Some features and lessons

We strongly believe that research on Intelligent Tutoring Systems [1] [2] will progress from both new theoretical models and various developments of applications. Our aim in this paper is to report about the design of SAIDA [3], an experimental environment using artificial intelligence techniques to teach a computer science course on abstract data representations. We focus on some specific functionalities our system provides to the different categories of users and we point out questions drawn from this experiment for further research.

The application domain is that of implementing algorithms with data types into efficient programs. The target students are in service or future computer science engineers. We explain in the first part why a knowledge base approach is particularly suitable for this domain and we briefly describe the architecture of the system.

Most tutoring systems are not yet flexible enough and do not take the different actors of education environments into account. We describe a system in which we identify several types of actors and provide each type with functionalities suitable for the required functions. This typology of actors in SAIDA is described in part two.

From the Artificial Intelligence point of view, one of the main components in these systems is the knowledge base. We explain in the third part the importance of a meaningful structure of the knowledge domain and we discuss several issues to build knowledge bases for educational purposes. Suitable automated or interactive tools are also mentionned.

I - Objectives and description of the SAIDA environment

SAIDA is an Help System for Data Implementation in ADA language including a knowledge base and appropriate library packages.

1) Why a knowledge-based system for advanced programming concepts learning and teaching ?

Computer Assisted Instruction for computer science teaching is mainly concerned with program construction for beginners [1] [2]. There are needs for environments dealing with advanced programming concepts. Among these advanced concepts, data abstraction and efficient representation choices seem to be of growing importance for at least the following reasons.

Books on data types mostly present a catalog of possible representations for each type and efficient algorithms to implement operations on these representations. Some of them are illustrated

by school-problems which do not enable students to learn criterions for appropriate choices in the algorithms they are building for solving real problems.

In real problems, several data structures are to be represented and efficient choices are often resulting from compromises between conflicting solutions and shadow reasoning on external constraints and intrinsic features of the involved algorithm. We have observed for a long time the difficulties of students carrying on such a task and we hope SAIDA will help them.

Was it worthwhile to gather this programming expertise and to build a knowledge base on this topic ? We have been induced by the growing importance of ADA in software development places and the related need for training a lot of programmers to the concept of data abstraction and efficient representation choices for these data. ADA allows to write procedures at different levels of abstraction using a stepwise refinement process including choices for data representations by instanciation of library packages.

2) SAIDA main features

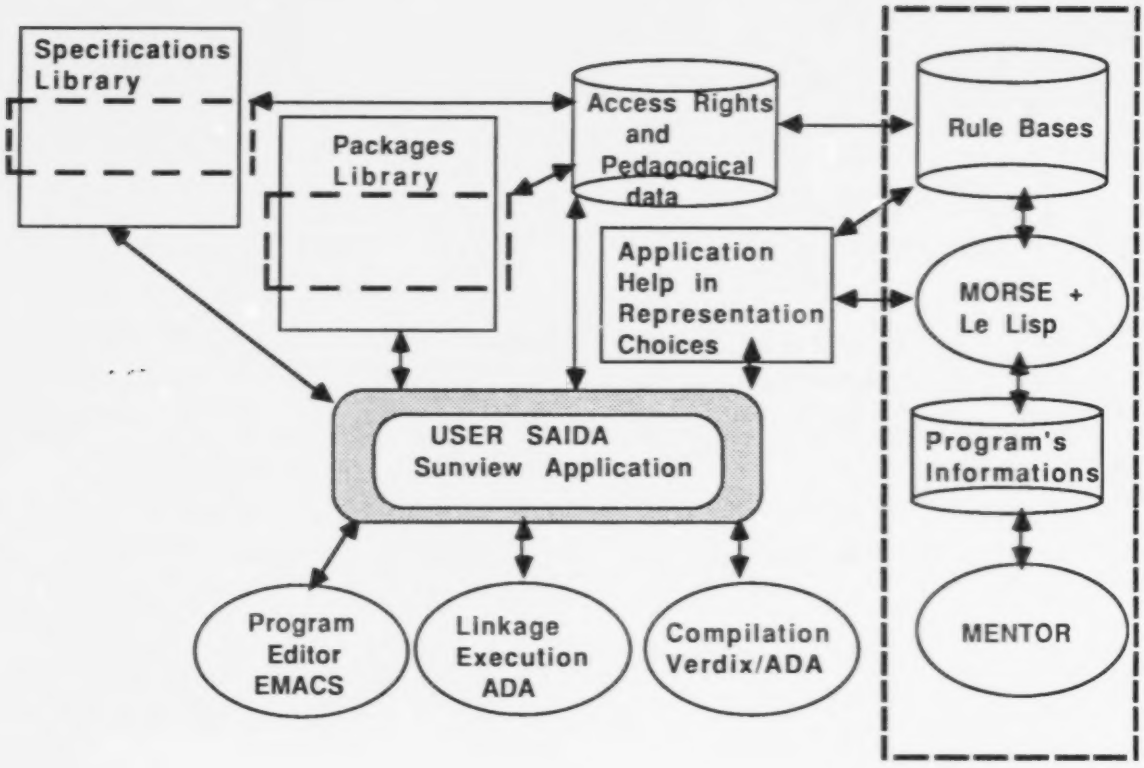
In the field of data representation for ADA programs, SAIDA helps to solve any problem brought by the student. The aim is not to communicate factual knowledge about representations, this is well done by a lot of books, but to allow the student to elaborate his own model of the task and his solving strategies by working with the expert system. Therefore SAIDA is an open system to every kind of problems.

When working with SAIDA, the student does not loose his familiar programming environment on his UNIX workstation. A windowing system allows him to run every program, consult every datafile. He is free to consult or not the expert system and to consult it as soon as he needs it. These features seem of very importance for non-beginners who need to keep the initiative in their tasks organisation.

So we can say that SAIDA is an open learning environment for the student. But it is also an open teaching system for the teacher. The teacher may choose different reasoning strategies depending on what he knows about his students abilities. He may enrich or modify the environment picking up what he wants in the provided libraries or building new modules by himself and thus create a pedagogical environment well suited to his students and to his own pedagogical project.

3) SAIDA architecture

The general architecture of the system is described on the following figure. A prototype has been running on a SUN3-DATA SYSTEME workstation since June 1988 at CRIN (Research Center in Computer Science, Nancy).



Several software are cooperating under UNIX system for building the SAIDA environment. Among them let us describe :

- Sunview, a windowing manager laying on a library of procedures written in C, the top level piloting of the application is written with it,

- Verdix/ADA and its libraries, a programming environment used to run the ADA programs built by students,

- Mentor/ADA, a syntactical analyser in order to automatically acquire informations on the procedure containing the complex data for which a representation choice has to be made,

- Morse, a rule-based expert system generator written in Le_lisp.

The system is now tested and will be improved with users remarks. One may find a more detailed presentation of SAIDA in [4].

II - The actors

We have just described SAIDA as an environment open to the student and to the teacher. This simplified view of the system's users does not fit the complex reality of artificial intelligence tutoring systems. General models of learning environments such as PETE [5] identify much more main actors. We would like to explain here what actors we need to identify in SAIDA and what functionalities are provided for them.

1) Actors during design and modification stages

- The system's author : for defining the aims of the system, choosing its architecture and piloting the realisation,

- the domain expert : for providing domain knowledge expertise and designing the resulting domain knowledge base ; in SAIDA we would need the domain expert again for instance to extend the actual base to new data types not yet included,

- the computer science teacher : he is the domain expert, but moreover he has pedagogical knowledge about beginners' difficulties in the field, problem-solving strategies better-suited to specific categories of learners... he writes the pedagogical part of the knowledge base ; this is implemented with the concept of steps in SAIDA,

- the programmer : he may help to produce packages to fill the system's programs library.

Other actors would be a psychopedagogue for general teaching strategies and a multimedia specialist for pictural or sound devices, but we did not need them for our application.

2) Actors during the use stage

- The configuration's author : he may configurate a version of SAIDA appropriate for the needs of a group of students (some data types and some implementations stored in the library, some problem-solving available strategies) ; he thus extracts an appropriate subsystem from the whole system provided,

- the teacher : he works on a configuration defined by the configuration's author, he identifies his students on the system and may ask for specific lays-out,

- the group of learners : they share the same version of the environment, which is in fact defined by their pedagogical profile (prerequisites, learning style...). In SAIDA this is implemented by access rights. This feature allows the system to be used either in a perspective of individual learning or with collective approaches,

- the learner : he is recognized by the system through his name and category , he may find and create his own files, programs and libraries,

- the operating system engineer : he manages common ressources such as available space on disk, passwords...

All these actors are not specified in other tutoring systems ; we believe they all need to be provided with suitable tools. Among these tools, those allowing to build and work on the knowledge bases of these systems are of crucial interest and we explain why in the next section.

III - Knowledge bases : design and use

1) The domain knowledge base in SAIDA

Two kinds of knowledge are represented in SAIDA : domain knowledge about data types representation criterions and knowledge describing solving strategies for domain problems. Let us focus here on domain knowledge. Clancey in [1] points out the importance of the internal structure of the domain knowledge base : "the author's choice of goals in the program constitutes a decomposition of the problem into reasoning steps".

We first organise domain knowledge into levels of concepts and identify main steps in the problem solving process. Then we write our rules using this framework. Domain concepts and steps in reasoning are then linked to express problem solving strategies. Such strategies are dedicated to specific pedagogical procedures such as exploring all possible situations for beginners or going straight forward to the goal for advanced students. Pedagogical rules allow choices between strategies.

We think that in the expertise transfer paradigm, good domain structuring and appropriate abstraction levels will help the student to create his own abstractions, his own domain model in which he will be able to solve problems alone. So we pay a great attention on displaying domain concepts, reasoning steps and problem solving thread.

2) What knowledge bases for pedagogical environments ?

The design and effective building of a domain knowledge base is a costly task, so it seems a good choice to use an existing base as often as possible. Such a knowledge base may be a component of a pedagogical system if at least the following conditions are satisfied : the knowledge domain is well structured and the structure appears clearly in the knowledge expression, this structure is a suitable basis for a learning process, the expert reasoning processes are to be learnt and used by the student. If these conditions are satisfied, it is possible to build a pedagogical level around this domain base [1].

For other domains, for instance technical areas, knowledge bases exist but are inappropriate for teaching purpose : knowledge is expressed in terms inadequate for learners or without apparent structuration. They must be reorganized, the knowledge bases have to be written again but without the contribution of an expert.

In other cases, professionnal bases do not exist because the domain is typically a pedagogical one as in plane geometry for 12-15 years pupils. In this case, a knowledge base has to be built and we think such bases would be designed in such a way that they constitute a basis tool for a lot of pedagogical applications. Once again, building a knowledge base is a costly investment,

it has to be done on a large scale so as to be reused by a large educational community.

We think that the lack of such bases in very different teaching areas will be a bottleneck in the development and the large scale experimentation of educational knowledge based systems.

3) What tools to build and use knowledge bases ?

Most needs are not specific to learning applications, but they are at least as crucial as in other domains if we want educators to develop such systems (different knowledge representation modes, friendly editors..).

Someones appear to be very usefull for educational environments. During the learning process, the student cannot learn all the domain concepts at once. So, it is often necessary to give him the part of the knowledge base dealing with the concepts he knows and increase step by step his view of the knowledge base until he is able to use and understand it entirely. How to extract a "sub-base" from a given base ensuring problem solving in a restricted universe ? It is easily realised if different concepts are not connected together in the reasoning process. But in other cases, we think a tool would be very interesting for teachers wanting to adapt a given knowledge base to their own teaching process.

SAIDA lays out a lot of interesting features even if all suitable functionalities are not yet implemented. Its different actors enrich a general model for Intelligent Tutoring Systems. It points out questions on knowledge bases which require more theoretical developments and further researches.

References :

[1] G.P. KEARSLEY Artificial Intelligence & Instruction, Applications and Methods, Addison-Wesley, 1987

[2] J.F. NICAUD, M. VIVET Les tuteurs intelligents, réalisations et tendances de recherche, TSI, 7 n°1, Dunod, 1988

[3] M. GRANDBASTIEN Towards a better use of computers in computer science teaching, Lausanne ECCE 88, North Holland 1988

[4] M. GRANDBASTIEN Une approche à base de connaissances pour l'enseignement de la programmation, Thèse d'Etat Université de Nancy, 1988

[5] DELTA Workplan, Commission of the European Communities, 1988